Datenbanksysteme WS 2016/17
Prof. Dr.-Ing. Sebastian Michel
BSc. Peter Brucker
TU Kaiserslautern, FB Informatik – Lehrgebiet Informationssysteme
http://dbis.informatik.uni-kl.de
Übungsblatt 2: Ausgabe 08.11.2016, Abgabe 14.11.2016

Aufgabe 1: Analyse Seitenreferenzstrings (1 P.)

(a) Gegeben folgender Seitenreferenzstring:

⟨(T1,A) (T2,B) (T1,A) (T1,A) (T2,C) (T2,D) (T1,D) (T1,A) (T1,G)
(T2,F) (T2,F) (T2,G) (T2,O) (T1,A) (T1,B) (T2,D) (T1,D) (T2,E)⟩

Berechnen Sie Sequentialität und Lokalität. Für Lokalität verwenden Sie die Working-Set-Methode und betrachten Sie den Durchschnitt über den beiden Transaktionen.

(b) Gegeben folgende Stacktiefenverteilung. Beantworten Sie folgende Fragen:

i) Wir möchten nicht mehr als 10% Cache-Misses haben. Wie groß muss der Puffer sein?

ii) Wie viele Cache-Hits (in Prozent) gibt es bei einer Puffergröße von 3?

iii) Wie viele Cache-Misses werden vermieden, wenn der Puffer von 3 auf 10 vergrößert wird?

Stacktiefe	Häufigkeit
1	411
2	112
3	84
4	60
5	54
6	33

Stacktiefe	Häufigkeit
7	21
8	14
9	5
10	4
11	2
12	1

Aufgabe 2: Abfragekosten (1 P.)

Gegeben eine Relation R mit Primärattribut a. Vier Tupel dieser Relation passen in eine Seite, jedes Tupel besteht aus vier gleich großen Attributen und `select count(*) from` R ergibt 200.000. Alle Indexe sind B^+-Bäume mit Höhe h, deren Blattknoten acht Einträge mit Verweisen auf Seiten fassen. Die Blattknoten sind komplett gefüllt. Der Datenbankpuffer ist vernachlässigbar.

(a) Geben Sie die Kosten in Seitenzugriffen für die Abfrage `select * from` R `where` p_a bei Verwendung des Primärindexes an. Dabei ist p_a eine Bereichsanfrage über das Primärattribut, die 25% der Daten auswählt.

(b) Was würde `select * from` R `where` p_b kosten, wenn der Optimierer sich für einen Sekundärindex über dem Nicht-Primärattribut b entscheidet? p_b ist wieder eine Bereichsanfrage über b, die 25% der Daten auswählt (also z.B. b`>=25 AND` b `<=50`, wenn b gleichverteilte Werte zwischen 1 und 100 annehmen kann).

(c) Sei nun `select` b `from` R `where` p_b die Anfrage nach b. Was kostet diese Anfrage bei Verwendung eines Sekundärindexes über b?

(d) Was kostet die selbe Anfrage mit dem selben Index, wenn die Tabelle zuvor nach b geclustert wurde?

Aufgabe 3: Join-Implementierungen (1 P.)

(a) Block Nested Loop Join I

Gegeben sind die Relationen *User* und *Comment*, über die ein Block Nested Loop Join (ohne Hashtabelle) durchgeführt wird, bei welchem ein größerer „Chunk“ an Seiten auf einmal übertragen wird. Es gibt 2286 Benutzer und 13970 Einträge in der Kommentar-Tabelle. Die Tupel beider Relationen sind jeweils eine Seite groß, der zur Verfügung stehende Join-Puffer fasst 256 Seiten.

- Welche Größe hat ein Chunk der äußeren Relation?
- Berechnen Sie die Anzahl der Seitenzugriffe auf die Festplatte für beide Möglichkeiten der Wahl der äußeren Relation.

(b) Block Nested Loop Join II

Betrachten Sie außerdem eine alternative Implementierung, bei der der Join-Puffer gleichmäßig mit Seiten der inneren und äußeren Join-Tabelle gefüllt wird.

- Wie müssen Sie das Kostenmodell anpassen?
- Wie hoch sind die Kosten, die in ihrem Model beim Join zwischen *User* und *Comment* verursacht werden?

(c) Grace Hash Join

Gegeben zwei Relationen R und S, die wie folgt verlustfrei gejoint werden sollen: $R.a \bowtie S.b$. R habe 100.000 Tupel mit fortlaufenden Werten von 1 bis 100.000 für a, S habe 500.000 Tupel mit gleichverteilten Werten für b. Die Breite der Tupel sei so groß wie eine Seite. Der Join-Puffer habe Platz für 80 Seiten wobei beliebig viel zusätzlicher Platz für eine Hashtabelle ist. Als Hashfunktion für die Partitionierung wird $\bmod k$ verwendet, gehen Sie davon aus, dass $a \subseteq \mathbb{N}, b \subseteq \mathbb{N}$.

- Wie wird k am geschicktesten gewählt, um den Grace Hash Join möglichst effizient auszuführen?
- Für Ihr k, wie viele lesende Seitenzugriffe sind nötig?
- Aus wie vielen sequentiellen Leseoperationen setzten sich diese zusammen?
- Wie viele Seitenzugriffe sind das verglichen mit dem Block Nested Loop Join für beide Möglichkeiten, die äußere Relation zu wählen?

Aufgabe 4: Zusatzaufgabe: Dies und Das (0 P.)

Diese Aufgabe soll Ihnen beim Verständnis der Themen Hilfestellungen geben. Sie wird nicht in der Übung besprochen und wir veröffentlichen keine Musterlösung.
Hinweis: Derartige Aufgaben sind auch immer Bestandteil der Klausur, sie sind daher auch gut als Vorbereitung derselben geeignet.

(a) Beschreiben Sie, was Seitenreferenzstrings sind und wie man Lokalität berechnen kann.

(b) Begründen oder widerlegen Sie folgende Aussage: Ein Index auf einem Nicht-Schlüssel-Attribut kann kein Clustering-Index sein.

(c) Begründen oder widerlegen Sie folgende Aussage: Ein Composite-Key-B+-Baum-Index der Relation R auf Attributen (a,b) kann von Nutzen sein für eine Anfrage der Form `select b,a from R where a < 100`.

(d) Diskutieren Sie Eignung von Clustering-Index, Non-Clustering-Index und Sortierung für sortierten Zugriff auf Daten.

(e) Wieso geht man davon aus, dass Datenbankpuffer überhaupt einen Nutzen haben?

(f) Erklären Sie, wieso es generell keine gute Idee ist, für alle möglichen Attribute und deren Kombinationen einen Index anzulegen.

(h) Ist es möglich, zwei Clustering-Indexe auf der gleichen Relation zu haben? Begründen Sie ihre Antwort.

(i) Erläutern Sie den Unterschied zwischen Primär-Index, Clustering-Index und sekundärem Index.

(j) Zum Zeitpunkt t habe eine Seite die Wahrscheinlichkeit b, dass auf sie zum Zeitpunkt $t+1$ zugegriffen wird. In wie vielen Zeitschritten wird voraussichtlich wieder auf die Seite zugriffen und wieso? Und was hat dies mit LRU zu tun?